**Cryptocurrency Dashboard Project Documentation**

**1. Introduction**

**Project Title: Cryptocurrency Dashboard**

**TEAM MEMBERS:**

* **Team Leader:M.Gnanesh**
* **Team Member:B.Nambeeswar**
* **Team Member:k.Deepak**
* **Team Member:** Y.Kishor
* **Team Member:** S Mahendran

A cryptocurrency dashboard that displays historical price data over the past five years is a powerful tool for investors seeking a comprehensive understanding of market dynamics.

**2. Project Overview**

**Purpose:**

The goal of this project is to build an interactive and user-friendly dashboard that aggregates and visualizes cryptocurrency data, allowing users to track prices, monitor portfolio performance, and stay updated with market trends.

**Features:**

* **Real-Time Price Tracking:** Displays up-to-date prices for various cryptocurrencies.
* **Portfolio Management:** Allows users to manage their cryptocurrency investments and track portfolio performance.
* **News and Alerts:** Provides cryptocurrency-related news and custom alerts for price changes or market events.

**3. Architecture**

**Component Structure:**

The main React components are structured as follows:

* **App:** Root component that includes routing logic and global context.
* **Dashboard:** Displays real-time cryptocurrency prices.
* **Portfolio:** Allows users to manage their cryptocurrency holdings.
* **NewsFeed:** Displays cryptocurrency news and user-defined alerts.

These components interact through props, and state management is used to share data across them.

**State Management:**

We are using **React Context API** to handle global state management, which allows us to share important data like cryptocurrency prices and portfolio information across components without passing props down manually.

**Routing:**

**React Router** is used for navigation. The application has separate routes for the Dashboard, Portfolio, and News pages. The routing structure is simple:

* / - Dashboard page.
* /portfolio - Portfolio page.
* /news - News and Alerts page.

**4. Setup Instructions**

**Prerequisites:**

* **Node.js:** Ensure you have Node.js installed. You can download it from [Node.js website](https://nodejs.org/).
* **npm:** Node Package Manager (npm) is required for managing dependencies.

**Installation:**

1. Clone the repository:
2. git clone https://github.com/your-repo/cryptocurrency-dashboard.git
3. cd cryptocurrency-dashboard
4. Install dependencies:
5. npm install
6. Configure environment variables:
   * Create a .env file in the root directory.
   * Add the necessary API keys for cryptocurrency data sources.
7. Run the app locally:
8. npm start

**5. Folder Structure**

**Client:**

* **/components:** Contains reusable UI components like buttons, charts, and cryptocurrency cards.
* **/pages:** Contains the views/pages for the app (Dashboard, Portfolio, News).
* **/assets:** Stores images, icons, and other static resources.
* **/hooks:** Custom hooks for data fetching and managing portfolio state (e.g., useFetchData, usePortfolioState).
* **/styles:** Contains global and component-specific CSS.

**Utilities:**

* **/utils:** Helper functions and utility classes (e.g., API calls, formatting data).

**6. Running the Application**

To run the application locally, execute the following command:

npm start

This will start the development server, and you can view the application at http://localhost:3000.

**7. Component Documentation**

**Key Components:**

* **App.js:** The root component that includes routing and global state management.
* **Dashboard.js:** Displays the list of cryptocurrencies and their current prices.
  + **Props:**
    - cryptos: List of cryptocurrencies fetched from the API.
* **Portfolio.js:** Allows users to add, remove, and view their cryptocurrency investments.
  + **Props:**
    - portfolioData: Data of the user's portfolio.
* **NewsFeed.js:** Displays the latest cryptocurrency news.
  + **Props:**
    - news: List of news articles related to cryptocurrencies.

**Reusable Components:**

* **CryptoCard.js:** A reusable component to display individual cryptocurrency details like price, volume, and market cap.
* **AlertModal.js:** A reusable component for showing alerts (price changes, news updates).

**8. State Management**

**Global State:**

Global state is managed using **React Context API** to allow seamless sharing of data across the app:

* **cryptocurrencyData:** Stores real-time prices and related data of cryptocurrencies.
* **portfolio:** User's investment portfolio data.
* **alerts:** Stores the user's price and news alerts.

**Local State:**

Local states are used within components to handle specific UI states such as input values, loading states, and modal visibility.

**9. User Interface**

**Screenshots or GIFs:**

* Screenshot of the **Dashboard** displaying real-time prices of cryptocurrencies.
* GIF showing how users can manage their portfolio by adding or removing coins.

(Include relevant screenshots or links to GIFs here)

**10. Styling**

**CSS Frameworks/Libraries:**

* **Styled Components** is used to style React components.
* **CSS Modules** are utilized for component-specific styles to avoid global namespace collisions.

**Theming:**

The app supports light and dark modes. The user can toggle between these themes using a button in the UI. Theming is handled via a custom theme context and applied using styled-components.

**11. Testing**

**Testing Strategy:**

* **Unit Testing:** Use **Jest** for unit tests to validate individual functions and components.
* **Integration Testing:** **React Testing Library** is used to test the interaction between components and their integration.
* **End-to-End Testing:** **Cypress** is used for simulating user interactions and ensuring the entire app works as expected.

**Code Coverage:**

We use **Jest** along with **Istanbul** for measuring code coverage, aiming to achieve at least 80% coverage for key components and features.

**12. Screenshots or Demo**

You can view the demo of the app at: [Demo Link](https://demo-link.com/).

(Screenshot or demo link)

**13. Known Issues**

* **Data Delays:** Cryptocurrency price data may experience delays due to API rate limits or server issues.
* **Responsive Issues:** Minor layout issues when viewing on smaller screens (e.g., mobile).
* **Portfolio Sync:** Occasionally, portfolio data may take a moment to sync, especially when switching between devices.

**14. Future Enhancements**

* **Mobile App:** Develop a mobile version of the dashboard to make it accessible on the go.
* **Advanced Charts:** Implement advanced charting features for better visualizations of price trends.
* **Multi-Currency Support:** Allow users to view prices in different fiat currencies like USD, EUR, GBP, etc.
* **Push Notifications:** Implement push notifications to alert users about price changes or news events.

This format provides detailed documentation for your cryptocurrency dashboard project. Feel free to customize it according to the specifics of your project as necessary!